HOW TO CASTf90

Sabine Radanovics

August 11, 2016

Contents

1 What does it do? 1

2 Requirements 2

3 Get the code 2
3.1 Get the code from github... 2
3.1.1 You want to update to the latest version? 2
3.1.2 You want to contribute to the development of CASTf90? 2
3.2 ...or download the official version from the A2C2 project homepage 3
3.3 List of scripts and source files 3

4 Compile the program 4
4.1 On obelix/with the ifort compiler 4
4.2 Local installation with gfortran including the lapack installation and the netcdf fortran90 interface 4
4.2.1 Complete local installation compiling all dependencies . . 4
4.2.2 Custom local installation, for example if some of the libraries are already somewhere on your system or you plan to re-use the libraries in other programs 4

5 Run CASTf90 7
5.1 Run including the whole data handling process 7
5.2 Batch system 10
5.3 Run calculation only (you already have the data files and a corresponding configuration file) 10
5.4 Run as a Birdhouse process 12

6 Output 12

7 Legal Issues (Disclaimer) 12
7.1 Liability 13
7.2 Warranty 13

1 What does it do?

CASTf90 first downloads fields from NCEP reanalysis (sea level pressure, slp, as default) and then searches for a given simulation period the most similar cases

within a given data base period according to a given distance measure. Finally it writes the N most similar days including the calculated distances for them to an output file.

2 Requirements

CASTf90 is developed for Linux systems. It might run on macOSX, but this was never tested.

CASTf90 requires:

- cdo (climate data operators)
- nco (netCDF operators)
- netCDF with Fortran90 interface
- a Fortran95 compiler
- lapack95 libraries

3 Get the code

There are two possibilities to get the code:

3.1 Get the code from github...

1. Change to a directory of your choice.
2. Get a copy of the latest version in the repository

   ```
   git clone https://github.com/sradanov/castf90.git
   ```

   This will create a directory named "castf90", that contains all sources. The github repository does not only contain castf90 source code, but includes the source of the netcdf fortran90 libraries and the lapack95 library with its dependencies, that might be needed for a local install if not already installed.

3.1.1 You want to update to the latest version?

1. change to the castf90 directory
2. run

   ```
   git pull
   ```

3.1.2 You want to contribute to the development of CASTf90?

1. Create an account on github (if you don't already have one)
2. Write an email with your github username to sabine.radanovics@lsce.ipsl.fr in order to get added as a developer and thus get permissions to push your changes to the repository.

3.2 ...or download the official version from the A2C2 project homepage

https://a2c2.lsce.ipsl.fr/index.php/licences

You can download the complete lapack95, lapack and blas library code from
http://www.netlib.org/lapack95/ file lapack95.tgz
http://www.netlib.org/blas/blas.tgz
http://www.netlib.org/lapack/lapack.tgz

3.3 List of scripts and source files

After checkout or download the following files should be present:

Shellscripts

- run_analogs_case.sh
- getNCEP_slp.sh
- retrieve.sh

Source files

- analogue.f90
- config.f90
- distance.f90
- eofs.f90
- read.f90
- routines.f90

Makefile

- Makefile
- Makefile.home
- install.sh

Example pbsscript

- pbsscript

Library code (github only)

- Directory: Blas
- Directory: Lapack
- Directory: Lapack95
- Directory: Netcdf_fortran

Manual

- howto.pdf
- howto.tex

4 Compile the program

4.1 On obelix/with the ifort compiler

CASTf90 comes with a makefile to compile the fortran90 sources, however, it is assumed that netcdf libraries with fortran90 interface and the lapack library with fortran95 interface are installed. If you are working on "obelix" you just have to load the netcdf module using

```
module load netcdf/4p
```

before running

```
make -f Makefile
```

Elsewhere please make sure that netcdf **AND** its fortran90 interface are installed and that INCLUDEPATH in the makefile is the directory containing netcdf.mod, and LIBPATH the one containing libnetcdff.

The intel compiler already comes with the -mkl option that gives access to the lapack library, however, the lapack95 interface may need to be installed separately. INCLUDEPATH2 and LIBPATH2 contain the .mod and the lib files for lapack95 respectively.

4.2 Local installation with gfortran including the lapack installation and the netcdf fortran90 interface

4.2.1 Complete local installation compiling all dependencies

For a complete compiling all dependencies run

```
./install.sh
```

This will

1. compile the Blas library
2. compile the Lapack library
3. compile the Lapack95 interface
4. install the netcdf fortran libraries with fortran90 interface. This is the only part that is installed in the system using sudo (so you will be asked for a root password at some point).
5. compile CASTf90

.

4.2.2 Custom local installation, for example if some of the libraries are already somewhere on your system or you plan to re-use the libraries in other programs

Blas library There is a chance that there is already a file named libblas.so or libblas.a somewhere on your system. Probably in /usr/lib64 or something similar. If this is the case, you should be able to use it, as long as it had been

compiled with the same compiler version, and you can skip the "Compile the Blas library" paragraph. If not, there is probably a package in your package manager available (for example yum in the case of fedora) and you can install it using

```
sudo yum install blas
```

or something equivalent on other systems.

In case you want to compile the Blas library: Change to the directory containing the blas routines:

```
cd Blas
```

Compile the code and create a static library:

```
gfortran -O2 -c *.f
ar cr libblas.a *.o
```

Move the static library to a typical library directory such as /usr/lib64/. You will need the root password to do so.

```
sudo cp libblas.a /usr/lib64/
```

If you don't have the password, you may copy it to the main source directory where you will compile CASTf90.

Lapack As with Blas, there is a chance that there is already a file named liblapack.so or liblapack.a somewhere on your system. Probably in /usr/lib64 or something similar. If this is the case, you should be able to use it, as long as it had been compiled with the same compiler version, and you can skip the "Compile the Lapack library" paragraph. If not, there is probably a package in your package manager available (for example yum in the case of fedora) and you can install it using

```
sudo yum install lapack
```

or something equivalent on other systems.

In case you want to compile the Lapack library: Change to the directory containing the lapack routines:

```
cd Lapack
```

Open the file make.inc in the Lapack directory with your preferred text editor. At line 77-78 set the variable *BLASLIB* to the libblas.a (or libblas.so) you want to use, including its path!. Save the file. Run

```
make all
```

This will produce liblapack.a and libtmglib.a. Now you can move them to a directory of your choice. For convenience the same as libblas.a is located. For example:

```
sudo mv *.a /usr/lib64/
```

Lapack95 Lapack95 is an fortran95 interface for the lapack library, that is much more convenient to use, because it calculates the optimal workspace sizes that has to be given explicitly as arguments to the lapack routines. This limits the number of arguments to be passed to the subroutines and leads to more readable code because this avoids to declare the workspace size variables in the calling routine. This interface is likely not installed (except if you have used it already elsewhere).

How to compile the lapack95 interface? Change to the Lapack95 directory:

```
cd Lapack95
```

Open the file make.inc with your favorite text editor.

1. At line 17: Set the variable MODLIB to -I*path* where *path* is the path to the directory where you want to write the .mod files.
2. At line 23: Set the variable LAPACK_PATH to the directory (with path) where the libraries liblapack.a, tmglib.a and libblas.a can be found.
3. At line 25: Set the variable LAPACK95 to the path + filename the lapack95 library file should be written.
4. Save the file.

Change to the source directory:

```
cd Lapack95/SRC
```

Run:

```
make single_double_complex_dcomplex
```

This will produce the lapack95 library file (lapack95.a) and several module files (f77_lapack.mod, f95_lapack.mod, la_auxmod.mod, la_precision.mod). Again, lapack95.a can possibly be moved to the same directory as the other libraries (/usr/lib64/ for example), and the .mod files to an include directory, for example /usr/include/.

Netcdf libraries

1. Make sure that the basic netcdf libraries are installed (that there is a file libnetcdf.* in one of the library directories (e.g., /usr/lib64/). If not, it can likely be installed using the package manager (*sudo yum install netcdf* or equivalent).
2. Make sure that the netcdf fortran libraries are installed as well, including the f90 interface! This is the case if there is a file libnetcdff.* (yes, with 2 "f" in the end) is in a library directory (e.g., /usr/lib64/) **AND** a file named netcdf.mod in an include directory (e.g., /usr/include/). If this is not the case:
   (a) change to the Netcdf_fortran directory

```
cd Netcdf_fortran
```

(b) Set the NCDIR environment veriable to the directory including the libnetcdf.a or libnetcdf.so file

```
export NCDIR=/usr/lib64
```

or

```
setenv NCDIR /usr/lib64
```

depending on the active shell.

(c) run the configuration script with a path prefix for the lib and the include directory

```
./configure --prefix=/usr
```

(d) run

```
make check
```

(e) installation. For an installation in a directory other than /home/ an administrator password will be needed.

```
sudo make install
```

Compile CASTf90 Open the file Makefile.home in your preferred text editor.

1. At line 1: Set the variable INCLUDEPATH to the directory containing netcdf.mod.
2. At line 2: Set the variable INCLUDEPATH2 to the directory containing f95_lapack.mod and other lapack related .mod files.
3. At line 3: Set the variable LIBPATH to the directory containing libnetcdff.
4. At line 4: Set the variable LIBPATH2 to the directory containing liblapack95 and liblapack.
5. Save the file.

run

```
make -f Makefile.home
```

5 Run CASTf90

5.1 Run including the whole data handling process

Before running the script the first time or after an update, please verify that the variable *sourcedir* in run_analogs_case.sh (line 37) is set to the directory containing the source code.

To anable parallel calculation the OMP_NUM_THREADS environement variable has to be set. The command depends on the active shell, but might be something like

```
export OMP_NUM_THREADS=12
```

or

```
setenv OMP_NUM_THREADS 12
```

The number has to be adjusted according to the available ressources. When running through a batch system, this should be done in the batch script (see subsection 5.2 for details).

Then to start the procedure, run

```
./run_analogs_case.sh
```

run_analogs_case.sh accepts a number of options:

- Options to set I/O paths:
  - **-P**<*path to base data*>
  - **-p**<*path to simulation data*>
  - **-o**<*path to output file*>

  Paths have to end with a slash. Example:

  ```
  ./run_analogs_case.sh -P/home/scratch01/sradanov/A2C2/NCEP/
  ```

- Output file format option
  - **-f**<**file extension**> So far *.txt* (ascii) and *.nc* (netcdf4) are supported.
- Options related to the geographical region of interest:
  - **-R**<*region name*> Known region names are NA (North Atlantic) and NHmid (Northern Hemisphere midlatitudes). This region is used at the time of data download. Using a limited number of large regions for downloads avoids to store a version of the data for every domain one might want to use and to skip the download step if the data is already present. The -D option allows then to specify the spatial domain the analogues are actually calculated upon. New regions can be added in the script retrieve.sh from line 20 on.
  - **-D**<*lonmin*>,<*lonmax*>,<*latmin*>,<*latmax*> predictor domain, that is the spatial domain for which the analogues should be calculated. The domain has to be inside the region defined in option -R. Example:

    ```
    ./run_analogs_case.sh -RNA -D-20.0,50.0,22.5,70.0
    ```

- Time period selection:
  - **-S**<*YYYY-MM-DD*>,<*YYYY-MM-DD*> Simulation period
  - **-B**<*YYYY-MM-DD*>,<*YYYY-MM-DD*> Database/archive period (period from which the analogues are chosen)

Example:

```
./run_analogs_case.sh -S2013-12-01,2014-02-28 -B1950-01-01,1979-12-31
```

- Anomalie options
  - **-N**<*mode*> while <mode> is one of the following:
    - *none* analogues are calculated for raw fields.
    - *base* anomalies are calculated with respect to a smoothed seasonal cycle calculated from the database/archive period. The seasonal cycle is first calculated as day of year average over the years in the database period and then smoothed using a weighted moving average. For the smoothing window parameter see option -m.
    - *sim* anomalies are calculated with respect to a smoothed seasonal cycle calculated over the simulation period. (caution: make sure that the simulation is sufficiently long for meaningful cycle calculation)
    - *own* anomalies are calculated for each dataset (database and simulation) with respect to its own smoothed seasonal cycle. This option might be useful when simulation and database sets are from different sources and one want to get rid of the mean difference between the two data sets. (caution: change signals that manifest themselfs in the mean difference will probably be lost)
  - **-m**<*numberofdays*> number of days (preferably an odd number) to calculate weighted moving average for seasonal cycle smoothing. Weights are linearly decreasing towards both ends of the interval.

Example:

```
./run_analogs_case.sh -Nbase -m91
```

- Analogue calculation options:
  - **-v**<*varname*> Name of the NCEP field to download. The name has to be the same as in the filename in the NCEP database. Example for precipitable water:

    ```
    ./run_analogs_case.sh  -vpr_wtr.eatm
    ```

  - **-l**<*vertical level*> Either 'surface' for variables like slp or pressure level in hPa, e.g. '500'. Thge default is 'surface' Example for geopotential at 500 hPa:

    ```
    ./run_analogs_case.sh  -vhgt -l500
    ```

  - **-t**<*logical*> TRUE if the predictor variable (see -v) should be detrended, FALSE if not (default is FALSE). For example for geopotential as a circulation variable in order to remove the temperature induced trend.

- **-w**<*numberofdays*> Number of days of the year ± around the target day to consider as candidates. This parameter allows a seasonal restriction of the analogue search. For no seasonal restriction choose 183.
- **-d**<*distance*> Name of the distance to use for analogue calculation. Supported distances are (in increasing order of complexity):
  * *euclidean* (or equivalently *rms*, *rmse*)
  * *cosine* (or equivalently *cos*) Which is the cosine distance

$$dist = -\frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

  * *mahalanobis*
  * *of* (or equivalently *opticalflow*) Which is a displacement and amplitude difference similar to Keil and Craig (2009) derived using the optical flow field deformation technique (Marzban and Sandgathe, 2010). (Experimental...)

  Attention: the more complex the distance the longer the computation time. The computation time strongly depends on the domain size and the archive length!
- **-C**<*numberofdays*> Distances can be averaged over a *number of consecutive days* in order to find situations with similar time evolution.
- **-n**<*numberofanalogs*> Number of closest analogue dates to write to output.
- **-c**<*logical*> TRUE if rank correlation should be calculated as an additional diagnostic (written to the output file), FALSE if not.

Example:

```
./run_analogs_case.sh -w30 -dmahalanobis -C3 -n25 -cTRUE
```

- **-silent** This option reduces the things that are written to standard output.
- **-help** lists the options and default settings in the terminal and exits.

5.2 Batch system

If the run_analogs_case.sh script is submitted to a batch system (qsub command), the options are interpreted as options for the qsub command rather than as options for the run_analogs_case.sh script. A workaround is to run run_analogs_case.sh inside an other script which is then submitted to the batch system. *pbsscript* is an example for such a script.

5.3 Run calculation only (you already have the data files and a corresponding configuration file)

Suppose you have already constructed your input data files (and seasonal cycle files if needed), you can run the fortran program without the shell scripts if you provide a configuration file (textfile in fortran namelist form) like this:

```
&FILES
 my_files%archivefile = "/../../base_slp_NA_1985-01-01_2012-12-31_-80.0_50.0_22.5_70.0.nc"
 my_files%simulationfile = "/../../sim_slp_NA_1985-01-01_2012-12-31_-80.0_50.0_22.5_70.0.n
 my_files%outputfile ="/../../ana_slp_surface_rms_NA_sim_1985-01-01_2012-12-31_base_1985-0
 my_files%seacycfilebase = "/../../seasoncyc_base.nc"
 my_files%seacycfilesim = "/../../seasoncyc_sim.nc"
/
&PARAM
 my_params%timewin = 1
 my_params%varname = "slp"
 my_params%seacyc = .TRUE.
 my_params%cycsmooth = 91
 my_params%nanalog = 20
 my_params%seasonwin = 30
 my_params%distfun = "rms"
 my_params%calccor = .TRUE.
 my_params%silent = .FALSE.
/
&ATTS
 my_atts%simsource = "NCEP"
 my_atts%predictorvar = "slp"
 my_atts%archisource = "NCEP"
 my_atts%archiperiod = "1973-01-01,2012-12-31"
 my_atts%predictordom = "-80.0,50.0,22.5,70.0"
/
```

There are three groups: *&FILES*, *&PARAMS* and *&ATTS*. Each group contains some variables with values assigned. The values for the variables in *&FILES* are the file names including the full path for the archive file, the simulation file, the output file and the files containing the seasonal cycle to be removed from the archive and the simulation respectively. These are character chain type variables and thus the values need to be in double quotes. The variables have to be present even if no seasonal cycle should be removed - the values are ignored in this case though.

In the *&PARAMS* group:

- *my_params%timewin* (type integer) is the time window to average the score (see -C option)
- *my_params%varname* (type character) is the name of the variable to calculate the analogues on. Need to be the same as the variable name in the netcdf file.
- *my_params%seacyc* .TRUE. if seasonal cycle should be removed from the variable prior to the distance calculation, .FALSE. if not.
- *my_params%cycsmooth* (type integer) Smoothing paramenter (smoothing window in days) for the seasonal cycle. (see -m option)
- *my_params%nanalog* (type integer) Number of analogues (see -n option).
- *my_params%seasonwin* (type integer) Seasonal restriction (see -w option).

- *my_params%distfun* (type character) Distance function (see -d option)
- *my_params%calccor* .TRUE. for additional rank correlation output, .FALSE. otherwise. (see -c option)
- *my_params%silent* .TRUE. for silent mode, .FALSE. otherwise.

The *&ATTS* group is important if the output is written in netcdf format, since it allows to set some global attributes of the output file. All veriables are character chain type variables and thus the values need to be in double quotes.

Then run the program analogue.out with the configuration file as an argument.

```
./analogue.out configurationfilename
```

5.4 Run as a Birdhouse process

6 Output

The output is a multi-column text file containing:

- the date of the analyzed day (in the form yyyymmdd)
- the dates of the n best analogues (in the form yyyymmdd)
- the value of the minimised distance (to determine the analogues).
- the value of the spatial rank correlations between the n analogues and the analysed field if the -c option is TRUE.

The output file can serve as input for the AnaWEGE weather generator (Yiou, 2014).

7 Legal Issues (Disclaimer)

7.1 Liability

The owners liability shall not be incurred as a result of in particular (i) loss due the Licensee's total or partial failure to fulfill its obligations, (ii) direct or consequential loss that is suffered by the Licensee due to the use or performance of the software, and (iii) more generally, any consequential loss.

7.2 Warranty

You acknowledge that the scientific and technical state-of-the-art when the software program was distributed did not enable all possible uses to be tested and verified, nor for the presence of possible defects to be detected. In this respect, the your attention has been drawn to the risks associated with loading, using, modifying and/or developing and reproducing the software which are reserved for experienced users. You shall be responsible for verifying, by any or all means, the suitability of the product for your requirements, its good working order, and for ensuring that it shall not cause damage to either persons or properties. The owners hereby represents, in good faith, that they are entitled to grant all the rights over the software program. You acknowledge that the software program is supplied "as is" by the owners without any other express or tacit warranty, other than that provided for here above and, in particular, without any warranty as to its commercial value, its secured, safe, innovative or relevant nature. Specifically, the owners do not warrant that the software program is free from any error, that it will operate without interruption, that it will be compatible with your own equipment and software configuration, nor that it will meet your requirements. The owners do not either expressly or tacitly warrant that the software program does not infringe any third party intellectual property right relating to a patent, software or any other property right. Therefore, the owners disclaim any and all liability towards you arising out of any or all proceedings for infringement that may be instituted in respect of the use, modification and redistribution of the software program. Nevertheless, should such proceedings be instituted against you, the owners shall provide it with technical and legal assistance for your defense. The owners disclaim any and all liability as regards your use of the name of the software program. No warranty is given as regards the existence of prior rights over the name of the software program or as regards the existence of a trademark.

References

Keil, C. and Craig, G. C. (2009). A displacement and amplitude score employing an optical flow technique. *Weather and Forecasting*, 24:1297–1308.

Marzban, C. and Sandgathe, S. (2010). Optical flow for verification. *Weather and Forecasting*, 25:1479–1494.

Yiou, P. (2014). Anawege: a weather generator based on analogues of atmospheric circulation. *Geoscientific Model Development*, 7(2):531–543.